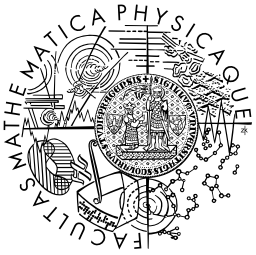


MATEMATICKO-FYZIKÁLNÍ FAKULTA
Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Zuzana Vopálková

Interaktivní vyhledávání v obrázkové kolekci pomocí neuronové sítě CLIP

Katedra softwarového inženýrství

Vedoucí bakalářské práce: doc. RNDr. Jakub Lokoč, Ph.D.

Studijní program: Informatika

Studijní obor: Programování a vývoj software

Praha 2023

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

Na tomto místě bych ráda poděkovala svému vedoucímu doc. RNDr. Jakubu Lokočovi, Ph.D. za neskutečnou ochotu, trpělivost a mnoho času, který byl ochotný věnovat celé této práci.

Chtěla bych také poděkovat všem respondentům, kteří mi ochotně poskytli svůj čas a umožnili tak získání dat pro experimenty v této práci.

Název práce: Interaktivní vyhledávání v obrázkové kolekci pomocí neuronové sítě CLIP

Autor: Zuzana Vopálková

Katedra: Katedra softwarového inženýrství

Vedoucí bakalářské práce: doc. RNDr. Jakub Lokoč, Ph.D., Katedra softwarového inženýrství

Abstrakt: S rostoucím významem a objemem multimediálních dat jsou nezbytné interaktivní vyhledávací systémy, které pomáhají uživatelům efektivně vyhledávat konkrétní videosekvence na základě obsahu. Jednou z běžných úloh je vyhledávání známé scény, kdy se uživatelé snaží najít konkrétní scénu v rozsáhlé kolekci videí. Popis známé scény však může být subjektivní, ovlivněný vnímáním a zkušenostmi jednotlivých uživatelů a také rozdíly mezi lidským a strojovým vnímáním. V této práci je k řešení tohoto problému zkoumána účinnost interaktivního vyhledávacího systému v kombinaci s klasifikací snímků generovanou pomocí neuronové sítě CLIP. K ověření účinnosti navrhované metody jsou použity datasety V3C a Marine Video Kit. Představen je i software, který pomocí webového rozhraní umožňuje sběr dat pro experimenty a jejich následné vyhodnocení.

Klíčová slova: interaktivní vyhledávání, CLIP, obrázkové databáze

Title: Interactive search in image datasets using CLIP neural network

Author: Zuzana Vopálková

Department: Department of Software Engineering

Supervisor: doc. RNDr. Jakub Lokoč, Ph.D., Department of Software Engineering

Abstract: With the growing importance and volume of multimedia data, interactive search systems are essential to help users efficiently search for specific video sequences based on content. One common task is known scene retrieval, where users try to find a particular scene in a large collection of videos. However, the description of a known scene can be subjective, influenced by the perception and experience of individual users, as well as the differences between human and machine perception. In this paper, the effectiveness of an interactive retrieval system combined with image classification generated by a CLIP neural network is investigated to address this problem. V3C datasets and Marine Video Kit are used to verify the effectiveness of the proposed method. Software is also presented that allows data collection for experiments and subsequent evaluation using a web interface.

Keywords: interactive search, CLIP, image databases

Obsah

Úvod

Multimediální data se v každodenním životě stávají stále důležitějšími, což demonstrují i neustále rostoucí objemy těchto dat.[23] Tento trend se pravděpodobně nebude ani měnit, neboť lidé neustále sdílí své zážitky prostřednictvím sociálních sítí.[1] To platí i v případě videí, které navíc nabízí velký potenciál využitelnosti v různých oblastech, od průmyslu přes zábavu až po vědecký výzkum. Nicméně s rostoucím objemem těchto dat bývá pro uživatele také náročnější efektivně nalézt konkrétní video sekvence na základě obsahu. Zde přichází na řadu *interaktivní vyhledávací systémy*[3, 14, 18, 22, 29, 34, 39], které uživatelům umožňují rychleji hledat konkrétní scénu i ve velkých kolekcích videí. Této úloze se obecně říká *vyhledávání známé scény* (known-item search).

Pro nalezení konkrétní scény nebo snímku je nejprve nutné nějakým způsobem zadat dotaz. K tomuto účelu se nejčastěji používají slovní popis nebo nějaké vizuální znázornění (nákres, barevné rozložení, ...) snímku. Tyto metody mají hlavní výhodu v tom, že jsou pro uživatele relativně známé a intuitivní. To znamená, že uživatelé nemusí být experty na daný systém multimediálního vyhledávání a i tak mohou snadno využívat rozhraní systému pro zformulování dotazu na požadovaný obsah. Nicméně popis známé scény je ovlivněn individualitou uživatelů (od způsobu přemýšlení přes jazykovou schopnost až po vnímání barev) a proto se způsob, jakým uživatelé popisují scénu, může výrazně lišit. Kromě rozdílu mezi uživateli vzniká rozdíl i mezi tím, jakým způsobem podněty v obrázku vnímají lidé a jakým způsobem je vnímá počítač (v případě této práce neuronová síť CLIP[24]). Tento rozdíl ve vnímání se někdy označuje jako tzv. *sémantická mezera* (semantic gap)[32, 11].

Problém různého vnímání scény může být řešen postupným zlepšováním samotných neuronových sítí. Na druhé straně je ale také možné uživateli přiblížit způsob vnímání sémantiky obsahu neuronovou sítí a tím zkusit minimalizovat sémantickou mezeru. To lze provést například zobrazením textových popisků snímků, které lze automaticky vygenerovat právě pomocí dané neuronové sítě. V této práci se zkoumá efektivita této metody s cílem nalézt vhodnou kombinaci interaktivního vyhledávání a zmíněné automatické textové anotace dat.

Pro zajištění větší nezávislosti výsledků experimentu na konkrétní kolekci dat byly pro experimenty zvoleny dvě různé domény – běžná video kolekce z internetu v podobě *V3C*[26] a kolekce zaměřená na podmořský svět v podobě *Marine Video Kit*[37] datasetu. Zvolení podmořské kolekce je důležité vzhledem k tomu, že uživatelé mohou mít omezenou znalost slovníku entit vyskytujících se na snímcích. Proto by na takové kolekci mohla být testovaná metoda více efektivní. Pro sběr dat od uživatelů (interakce a reformulace dotazů) byl vytvořen podpůrný software v podobě webové aplikace za použití frameworku *Django*[1]. Tento software sloužil také pro testování scénářů jako jsou způsoby připojování uživatelem zvolených tříd do textového dotazu při reformulaci nebo rozložení výsledků hledání. Kromě toho byl použit také pro výběr varianty neuronové sítě CLIP (ViT-B/32).

[1] www.statista.com/statistics/195140/new-user-generated-content-uploaded-by-users-per-minute/

1. Základní pojmy

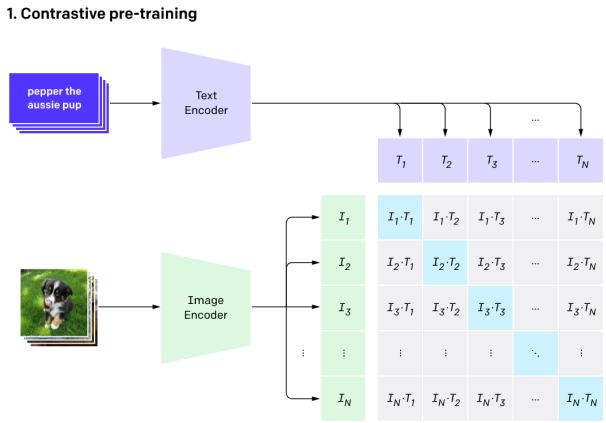
1.1 Vyhledávání známé scény

Problém týkající se vyhledávání ve videu je velmi komplexní a většinou se rozděluje na více podproblémů. Jedním z nich je vyhledávání známé scény (Known-Item Search neboli KIS). Tento problém tkví ve vyhledávání scény, která je uživateli známá v tom smyslu, že si pamatuje její obsah, avšak nezná název videa nebo čas začátku scény v konkrétním videu. V tomto případě je scéna ve větší kolekci videí velmi těžce nalezitelná pouhým procházením.

1.2 Neuronová síť CLIP

Nepostradatelnou součástí celé práce je neuronová síť CLIP (Contrastive Language–Image Pre-training)[24] vyvinutá společností OpenAI. CLIP využívá zpětnovazebního učení k asociovaní snímků s textem. Síť se skládá ze dvou enkodérů (znázorněných na Obrázku 1.1), přičemž první z nich je specializován na kódování textů a druhý snímků. Enkodéry mapují vstup na vysoko-dimenzionální reprezentaci v multimodálním prostoru. Díky tomu, že byl trénován na 400 miliónech párů (snímek, text), je CLIP použitelný pro široké spektrum domén.

Enkodovací funkce modelu CLIP budeme nadále označovat $f_{CLIP} : Img \rightarrow R^n$ pro snímky a $f_{CLIP\_W} : Text \rightarrow R^n$ pro text, stejně jako zde [17].



Obrázek 1.1: Postup společného trénování textového a obrazového enkodéru na sadě dvojic (snímek, text).[24]

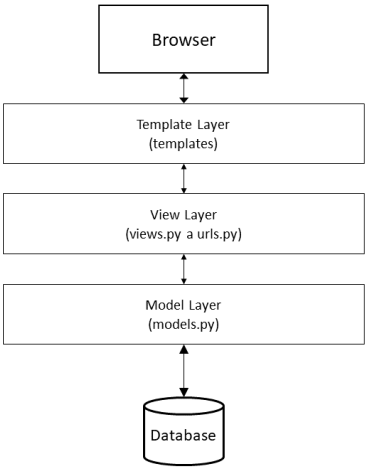
V podpůrném softwaru je využit model ViT-B/32 (Vision Transformer-B). Tento model byl vybrán hlavně kvůli jeho dobrým výsledkům při klasifikaci snímků na benchmarkových testech, jako je například *ImageNet*[6], v kombinaci s jeho velikostí, jelikož ViT-B/32 je menší z modelů (vstup je převeden na tensore o rozměrech 3x224x224 a výstupní tensor má velikost 512) v porovnání s ostatními CLIP modely (např. ViT-L/14 nebo RN50x4). Základem ViT-B/32 jsou dva paralelně trénované enkodéry, kterými jsou dvanáctivrstvý *Transformer*[38] pro transformaci textu a *Vision Transformer* (ViT)[7] pro zpracování obrazu.

1.3 Webový aplikační framework Django

Pro vytvoření celého podpůrného softwaru byl použit open-source webový framework *Django*[1, 9], který umožňuje vytváření webových stránek v Pythonu. Hlavním cílem knihovny je umožnit vývojářům snadný vývoj vysoce výkonných a bezpečných webových aplikací.

Důležitou výhodou Django je velké množství knihoven a nástrojů, jako jsou vestavěné administrátorské rozhraní, podpora pro autentizaci a autorizaci uživatelů, routování URL nebo správa databáze. Toto zajišťuje snadnou rozšiřitelnost a znovupoužitelnost kódu, a tedy usnadňuje samotný vývoj webu i s dodržením DRY (Don't Repeat Yourself) principu.

Architektura Django je založena na vzoru *Model-View-Template* (MVT), který je variantou obecnějšího vzoru *Model-View-Controller*[40]. Obecné schéma propojení vrstev je zobrazeno na Obrázku 1.2. Vrstva Model v Django je zodpovědná za definování datových modelů a správu interakce s databází. Používá *objektově-relační mapování* (ORM)[21] k poskytnutí vysokoúrovňového rozhraní pro interakci s databází. Vrstva View je zodpovědná za zpracování příchozích HTTP požadavků a rozhodnutí, jakou akci provést. Získává data z vrstvy Modelu, zpracovává je a vrací příslušnou odpověď klientovi. Views mohou být definovány jako funkce nebo třídy a mohou zpracovávat různé HTTP metody. Vrstva Template je zodpovědná za vykreslování HTML na základě dat předaných View.



Obrázek 1.2: Struktura Model-View-Template vzoru v Django projektu.

Další důležitou součástí architektury Django je URL dispatcher, který mapuje URL adresy na konkrétní funkce ve View vrstvě. Mapování URL adres je řešeno za pomoci 'urlpatterns' v 'urls.py' souboru, přičemž tento list definuje sadu regulárních výrazů, které Django použije na porovnání s příchozím URL a přiřadí pomocí tohoto seznamu funkci, která bude zavolána na zpracování aktuálního požadavku.

1.4 Interaktivní reformulace

Nejprve bychom měli definovat význam slova *reformulace*, které bude používáno po zbytek práce. Reformulaci používáme k označení nové verze dotazu vytvořené uživatelem v průběhu hledání jedné konkrétní scény v kolekci. Interaktivní reformulace s podporou systému umožňuje uživateli účinněji modifikovat dotazy a snáze nalézt požadované výsledky pomocí výsledků předchozích hledání. V případě *interaktivního vyhledávání*[28, 27] v obrázkové kolekci je možné proces interaktivní reformulace opakovat do nalezení hledaného snímku.

Samotný proces interaktivní reformulace je z velké části ovlivněn "parametry" samotného uživatele, jako jsou jeho zkušenosti, způsob přemýšlení nebo znalost dané domény. Zejména s posledním zmíněným faktorem lze pracovat pomocí nějaké formy nápovědy pro uživatele, která mu může pomoci s vyhledáváním. Jedním ze způsobů přiblížení domény a minimalizování tzv. sémantické mezery, zmíněné v úvodu, je poskytnutí metadat ke snímkům. Pro interaktivní textové vyhledávání je užitečné poskytovat popis, který může uživatel následně použít ve svém dotazu. Jednoduchou verzí automatické anotace snímků je klasifikace, která pro snímek vybírá jednu nebo více tříd z pevně dané množiny tříd (tzv. slovníku). Automatickou klasifikaci snímků umožňují neuronové sítě jako GoogLeNet[35], EfficientNet[36], nebo právě populární síť CLIP.

Využití interaktivní reformulace v kombinaci s poskytnutím klasifikace snímků může zlepšit celkovou efektivitu vyhledávání a pomoci uživatelům přesněji formulovat dotazy.

1.4.1 Základní algoritmus hledání

V této kapitole představíme *základní algoritmus vyhledávání v obrázkovém datasetu* (viz Algoritmus 1). Tento algoritmus je základem většiny později představených modelů. Základní algoritmus je založen na jednom z *klasických modelů*[31] pro vyhledávání informací, kterým je *model vektorového prostoru*.

Nejprve si zadefinujme značení vstupních parametrů, které bude používáno v průběhu celé této práce. Nechť Img je množina všech snímků, z nichž aktuálně používaný dataset snímků budeme značit D[1]. Platí tedy $D \subset Img$ a velikost datasetu budeme označovat n. I-tý snímek z datasetu D označíme p_i, kde i představuje index snímku a platí, že $i \leq n$. Hledaný snímek označíme H, kde $H \in D$, a primární uživatelský textový dotaz popisující aktuálně hledaný snímek H označíme q_0. V případě k-té reformulace textového dotazu reformulovaný dotaz označíme q_k. Nechť $f_{CLIP} : Img \to R^n$ a $f_{CLIP\_W} : Text \to R^n$ jsou funkce představující enkodéry modelu CLIP zmíněné v kapitole 1.2. U algoritmů bude nadále používáno podobné značení jako v této práci [17].

Základní myšlenkou modelu vektorového prostoru je reprezentovat dotaz a každý snímek z D jako vektor v n-rozměrném prostoru. V tomto modelu se dotazy a snímky zobrazují jako body v prostoru, kde souřadnice vektoru představují vlastnosti objektů. Pro zjištění relevance dotazu ke snímku se obvykle využívá kosinová vzdálenost mezi vektorem dotazu a vektorem snímku. Snímky s nejvyšší podobností (tj. nejmenší vzdáleností) jsou považovány za nejrelevantnější k danému dotazu. Pro zjednodušení výpočtu je možné vektory normalizovat

[1] Z důvodu přehlednosti v algoritmech označován jako *Dataset*.

tak, aby měly jednotkovou délku. V této práci budeme pro převedení snímků a dotazu do n-rozměrného prostoru používat enkódovací funkce modelu CLIP f_{CLIP} a f_{CLIP_W}.

Vstupem základního algoritmu hledání jsou textový dotaz q_0[2] a obrázkový dataset $D = |n|$. Nejprve pomocí funkce f_{CLIP_W} převedeme textový dotaz q_0 do n-rozměrného prostoru. Následně pro každý snímek p_i z datasetu D spočítáme kosinovou vzdálenost mezi textovým dotazem a vektorem reprezentujícím daný snímek ve stejném n-rozměrném prostoru, získaným pomocí funkce f_{CLIP}. Nakonec seřadíme snímky podle vzdálenosti od nejbližšího (neboli nejrelevantnějšího) snímku vůči dotazu.

Algorithm 1 $TextSearch(query, Dataset)$

Require: $f_{CLIP} : Img \rightarrow R^n$, $f_{CLIP\_W} : Text \rightarrow R^n$

$q \leftarrow f_{CLIP\_W}(query)$

for each $frame \in Dataset$ **do**

$frame.score \leftarrow \delta_{cos}(q, f_{CLIP}(frame.Image))$

end for

$SortByScoreAscending(Dataset)$

Získáme-li pozici snímku v celkovém setříděném výsledku, budeme tuto pozici nadále nazývat *rank snímku*. Tato pozice je indexována od 1, takže v případě, že je snímek ve výsledku na prvním místě, jeho rank je roven 1 a obdobně pro ostatní pozice.

1.5 Interaktivní hledání ve videu

V posledních letech se objevilo mnoho různých přístupů ke zpracování úlohy vyhledávání známé scény i z pohledu vývoje uživatelsky přívětivých interaktivních vyhledávacích systémů pro procházení obrovských vizuálních datasetů. Nové poznatky a technologie řešící KIS problém se představují každoročně například na mezinárodní soutěži *Video Browser Showdown* (nadále jen VBS)[30, 16], pořádané na Mezinárodní konferenci o multimediálním modelování (MMM). V této soutěži se porovnávají nástroje jednotlivých týmů v třech hlavních disciplínách, kterými jsou vizuální a textový KIS a ad-hoc hledání videa (ad-hoc video search neboli AVS). Všechny úkoly jsou dopředu vybrány z předem známých *kolekcí videí*[37, 26]. Při vizuálních KIS úkolech je soutěžícím zobrazen krátký úsek videa, který mají najít. Ve dvou dalších disciplínách je zobrazeno textové zadání, na jejichž základě se hledá jedna konkrétní scéna v případě textových KIS úkolů nebo všechny scény odpovídající zadání v případě AVS úkolů.

Další významnou událostí v tomto oboru je *Lifelog Search Challenge* (zkráceně LSC)[2], na které se soutěží nad rozsáhlým *datasetem snímků*[10] vytvořeném tzv. lifeloggingem, což je periodické pořizování snímků a dalších meta-dat v průběhu každého dne.

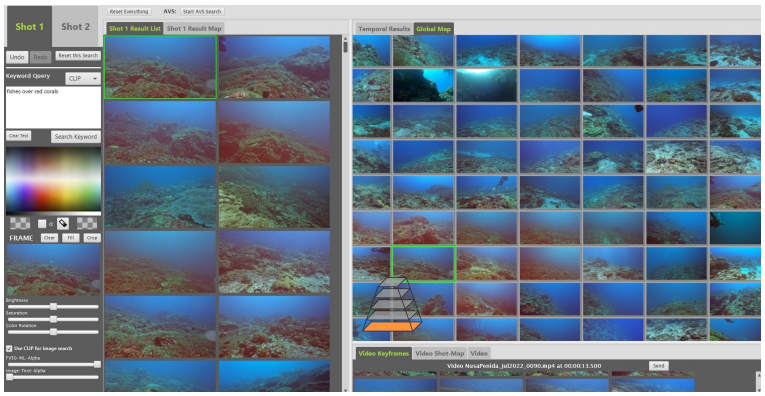
[2]Kvůli přehlednosti algoritmu 1 označovaný jako *query*.

1.5.1 Existující systémy

Následně si představíme některé z nástrojů, které dopadly nejlépe (v pořadí od prvního) na VBS2023 v Norsku, a obsahují některé z nejnovějších metod pro řešení KIS a AVS problémů.

HTW (Vibro)

Vibro[29] je interaktivní vyhledávač ve videu vyvíjený na univerzitě HTW v Berlíně. Vyhledávání je možné pomocí textu, skici, temporálních dotazů, podobnostního vyhledávání a prohledávání grafu podobností obrázků (image similarity graph)[12]. Výsledky dotazů jsou rozmístěny i na 2D mapě (možné vidět na Obrázku 1.3 v pravé části okna) za pomoci FLAS (Fast linear assignment sorting)[4].

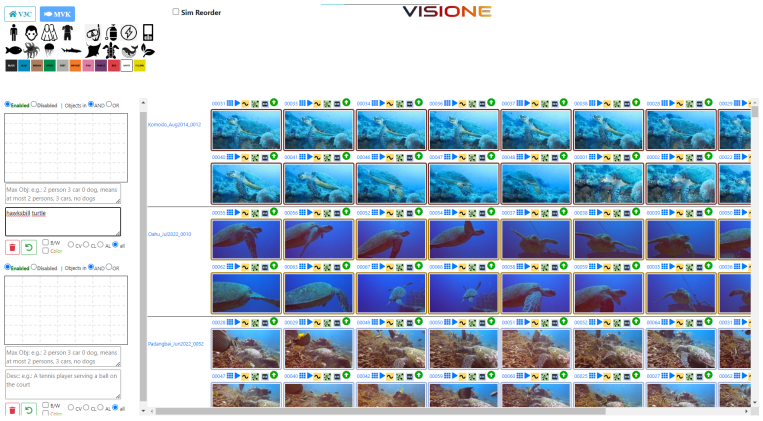


Obrázek 1.3: Ukázka softwaru Vibro.[3]

Visione

Visione[3] je systém na efektivní vyhledávání videí v rozsáhlých datasetech využívající "free text" vyhledávání, prostorové vyhledávání barev a objektů, vizuální a sémantické podobnostní vyhledávání a temporální vyhledávání. Na rozdíl od většiny nástrojů na VBS využívá Visione kromě CLIPu také modely CLIP2Video[8] a ALADIN (ALign And DIstill Network)[20].

[3]Tímto bych chtěla autorům nástroje Vibro poděkovat za poskytnutí snímku uživatelského rozhraní.



Obrázek 1.4: Ukázka uživatelského rozhraní nástroje Visione.[4]

Vireo

Vireo[18], stejně jako většina softwarů na vyhledávání ve videu, využívá vyhledávání pomocí textu, temporální dotazování a podobnostní vyhledávání. Kromě klasický metod Vireo integruje model z PicHunter[5] se zpětnovazebním učením k omezení rozsahu vyhledávání. Těmito technikami se systém snaží omezit počet iterací hledání.

vitrivr-VR

Vitrivr-VR[34] je rozhraní virtuální reality založené na Unity Engine pro multimediální vyhledávání používající vyhledávací engine Cineast[25]. Uživatelské rozhraní vitrivr-VR je schopné běhu na jakékoliv XR platformě podporující OpenXR. Pro vyhledávání využívá převážně slovní textové vyhledávání. Hlavní výhodou softwaru je do budoucna možnost hledat jen za pomocí brýlí a jednoduchých ovládacích prvků.

CVHunter

Context-aware Video Hunter[14, 17] navazuje na systémy SOMHunter[39] a VIRET[22] vyvíjené dříve na KSI MFF UK. Vyhledávání pomocí tohoto nástroje nejčastěji začíná pomocí zadání textového dotazu. Dále může uživatel využít podobnostní vyhledávání, interaktivní bayesovská zpětnovazební vyhledávání inspirované systémem PicHunter[5], temporální dotazování nebo sekvenční procházení reprezentativních snímků jednotlivých videí.

1.6 SOM

Samoorganizující mapa (Self-Organizing Map neboli SOM)[13] je typ neuronové sítě používající se na vizualizaci složitých vysoko-dimenzionálních dat v topologickém uspořádání pomocí redukce dimenze (obvykle na 2D). SOM je

[4]Autorům nástroje Visione bych chtěla touto cestou poděkovat za poskytnutí snímku uživatelského rozhraní a materiálů tykajících se jejich softwaru.

založena na modelu neuronové sítě složené z mřížky neuronů, z nichž každý reprezentuje určitý region vstupních dat.

2. Modely pro reformulaci dotazu

V této práci jsou zpracovány tři kategorie algoritmů na vyhledávání pomocí textového dotazu, které podrobněji představíme v této kapitole. Každá z těchto kategorií se zabývá jinou částí vyhledávání a proto je možné kombinovat algoritmy z různých kategorií dohromady. Všechny modely využívají základní rankovací algoritmus hledání bez reformulace (viz kapitola 1.4.1), který postupně rozšiřují o další varianty. V závěru kapitoly bude představen model, který se nezabývá přímo hledáním pomocí textového dotazu, ale umožňuje porovnat zkoumané modely s jiným způsobem vyhledávání v obrázkovém datasetu, kterým je v tomto případě podobnostní vyhledávání pomocí obrázku jakožto příkladu.

Algoritmy v této sekci, stejně jako základní rankovací algoritmus, kvůli jejich přehlednosti a srozumitelnosti neuvažují optimalizace jako je předzpracování všech vektorů snímků z datasetu.

2.1 Algoritmus hledání s reformulací - kombinace prvního a druhého dotazu

Tato skupina modelů je založena na různých formách fúze výsledků z primárního dotazu s výsledky reformulovaného dotazu. Celkem se v této skupině nachází pět základní modelů fúze, které využívají výsledky prvního dotazu, jež je vyhodnocený pomocí základního rankovacího algoritmu (viz Algoritmus 1). Pokud je s_1 hodnota kosinové vzdálenosti obrázku od prvního dotazu a s_2 vzdálenost od druhého dotazu, tak je možné tyto hodnoty kombinovat několika způsoby. První testovanou možností je sčítaní, druhou maximum, třetí násobení, čtvrtou minimum a pátou vážený průměr, kde výsledky druhého dotazu mají dvakrát tak větší váhu. Kromě těchto modelů do této skupiny patří i algoritmus, který využívá pouze novou reformulaci dotazu. Tento algoritmus sice nevyužívá přímo výsledky primárního dotazu[1], ale představuje zajímavou referenční metodu, která ukazuje vliv samotné reformulace. Předpokladem pro správné fungování tohoto algoritmu je, že druhý uživatelský dotaz je obecně lepší reprezentací hledaného snímku než první dotaz. Tento přístup může být hlavně užitečný v případě, kdy první dotaz byl nepřesný a jeho kombinace by zhoršila kvalitu výsledku reformulace.

K obecnému zadefinování těchto modelů využijeme funkci *combine*, která bude definovat aktuálně používanou fúzi výsledků dotazů. Do této funkce je kromě výsledků hledání předáván i parametr c, který určuje zvolenou fúzi (včetně využití pouze nového výsledku).

Vstupem algoritmu jsou primární dotaz q_0[2], reformulovaný druhý dotaz q_1[3] a obrázkový dataset D. Nejprve je provedena konverze obou textových dotazů do n-rozměrného prostoru za pomoci funkce f_{CLIP_W}. Následně jsou pro každý snímek z datasetu D spočteny dvě kosinové vzdálenosti. První jsou vzdálenosti mezi vektorem primárního dotazu a vektory snímků z D a druhé vzdálenosti

[1] Výsledky primárního dotazu jsou využity pouze uživatelem na vytvoření reformulovaného dotazu.

[2] Za účelem zvýšení srozumitelnosti algoritmu 2 označený jako *oldQuery*.

[3] Pro zlepšení přehlednosti algoritmu 2 označený jako *query*.

mezi vektorem aktuálního reformulovaného dotazu a opět vektory všech snímků. Tyto dvě vzdálenosti jsou následně zkombinovány pomocí obecné funkce *combine*. Toto celkové skóre je využito na následné vzestupné seřazení snímků od nejbližšího (nejpodobnějšího).

Algorithm 2 $ReformulationSearch(query, oldQuery, Dataset, c)$

Require: $f_{CLIP} : Img \to R^n$, $f_{CLIP\_W} : Text \to R^n$, $combine : R^2 \to R$

$q_{first} \leftarrow f_{CLIP\_W}(oldQuery)$
$q_{second} \leftarrow f_{CLIP\_W}(query)$

for each $frame \in Dataset$ **do**
$frame.firstScore \leftarrow \delta_{cos}(q_{first}, f_{CLIP}(frame.Image))$
$frame.secondScore \leftarrow \delta_{cos}(q_{second}, f_{CLIP}(frame.Image))$
$frame.overallScore \leftarrow combine(frame.secondScore, frame.firstScore, c)$
end for

$SortByOverallScoreAscending(Dataset)$

2.2 Algoritmus hledání s reformulací - omezování datasetu

V této části je představen model využívající omezení datasetu, ve kterém se vyhledávání provádí po reformulace. Tento algoritmus je založen na předpokladu, že dobrý prvotní dotaz neumožňuje výrazné oddálení hledaného snímku. Základem tohoto algoritmu je předem zadaný parametr $\gamma \in < 0; 1 >$, který definuje procentuální množství kolekce příliš vzdálené od původního dotazu, které se vynechá při reformulaci kvůli své nerelevantnosti. Největší motivací tohoto modelu je snížení potřebných výpočetních zdrojů na vytvoření výsledku reformulovaného dotazu právě díky omezení datasetu.

Algoritmus bere jako vstup primární dotaz q_0[4], reformulovaný (druhý) dotaz q_1[5], obrázkový dataset D a již zmíněný parametr γ, který je převeden na velikost již omezeného datasetu značenou *limitingSize*. Podobně jako u předchozích algoritmů jsou nejprve provedeny konverze obou textových dotazů do n-rozměrného prostoru za pomoci funkce f_{CLIP_W}. Následně je pro každý snímek z datasetu D spočtena kosinová vzdálenost mezi vektorem primárního dotazu a vektorem snímku. Dataset je poté seřazen podle vzdálenosti od nejbližšího snímku a omezen pomocí *limitingSize*. Pro každý snímek z tohoto omezeného datasetu je vypočtena kosinová vzdálenost mezi vektorem reformulovaného dotazu a vektorem snímku. Omezený dataset je seřazen podle nově vypočítaných vzdáleností od nejbližšího (nejpodobnějšího) snímku.

[4]Z důvodu přehlednosti algoritmu 3 označený jako *oldQuery*.
[5]S cílem usnadnit čitelnost algoritmu 3 označený jako *query*.

Algorithm 3 $LimitedReformulationSearch(query, oldQuery, Dataset, limitingSize)$

Require: $f_{CLIP} : Img \to R^n$, $f_{CLIP\_W} : Text \to R^n$

$q_{first} \leftarrow f_{CLIP\_W}(oldQuery)$
$q_{second} \leftarrow f_{CLIP\_W}(query)$

for each $frame \in Dataset$ **do**
    $frame.score \leftarrow \delta_{cos}(q_{first}, f_{CLIP}(frame.Image))$
end for

$SortByScoreAscending(Dataset)$

$newDataset \leftarrow Dataset[0 : limitingSize]$

for each $frame \in newDataset$ **do**
    $frame.score \leftarrow \delta_{cos}(q_{second}, f_{CLIP}(frame.Image))$
end for

$SortByScoreAscending(newDataset)$

2.3 Algoritmus hledání s inicializací pomocí SOM

Tento model přistupuje k problému vyhledávání poněkud odlišným způsobem a snaží se vylepšit již prvotní dotaz uživatele. K tomu využívá úvodní obrazovku inicializovanou za pomoci SOM, jejíž vstupními daty jsou vektory jednotlivých snímků, přičemž každý snímek je reprezentován jedním vektorem a tento vektor představuje třídy, do které byl snímek klasifikován. Třídy jsou ve vektoru reprezentovány jako indexy těchto tříd do slovníku (pevně daný seznam podstatných jmen).

Model založený na SOM je užitečný zejména pro uživatele, kteří nemají dostatečné znalosti názvosloví dané domény nebo si nejsou jisti, jaké vhodné termíny pro dotaz použít. Uživatelé mají díky SOM k dispozici větší škálu snímků, jejichž třídy mohou využít k formulaci již prvního dotazu. Model také umožňuje uživatelům upřesnit jejich dotaz na základě vybraných tříd a potenciálně tak získat z kolekce relevantnější obrázky.

2.4 Algoritmus na srovnání s hledáním pomocí snímku

V případě, že se hledaný snímek velmi podobá jednomu ze zobrazených snímků, může být efektivnější najít ho pomocí podobnostního hledání (algoritmus podobnostního hledání pomocí obrázku jako příkladu je vidět v Algoritmu 4) namísto použití modelů, které jsou zkoumané v této práci. Protiargumentem tohoto přístupu je fakt, že snímky nemusí být dostatečně globálně podobné, například

mohou obsahovat pouze stejný objekt, a v tomto případě by podobnostní hledání nemuselo být tak efektivní. Cílem tohoto algoritmu je proto porovnat tyto dva rozdílné přístupy k vyhledávání.

Algorithm 4 $SimilarityImageSearch(imageQuery, Dataset)$

Require: $f_{CLIP} : Img \rightarrow R^n$
$q \leftarrow f_{CLIP}(imageQuery)$

for each $frame \in Dataset$ **do**
$frame.score \leftarrow \delta_{cos}(q, f_{CLIP}(frame.Image))$
end for

$SortByScoreAscending(Dataset)$

Pro provedení podobnostního hledání je nutné získat snímek, na základě kterého bude vyhledávání provedeno. Abychom se vyhnuli organizaci další studie s uživateli, tak budeme tento výběr snímku pouze aproximovat na základě existujících dat z předchozích studií. Proto je potřeba zdůraznit, že jsou výsledky tohoto experimentu pouze orientační. Obrázek je označen jako dotaz pro potřeby tohoto experimentu vždy v případě, kdy uživatel přidá třídu snímku zobrazeného v aktuálním výsledku. Předpokládá se, že uživatel by tuto třídu mohl vybrat kvůli podobnostem mezi snímky, a proto by mohl být obrázek uživatelem vybrán i při podobnostním vyhledávání.

Pro porovnání podobnostního a textového vyhledávání je provedena retrospektivní analýza s využitím záznamů vybraných obrázků a původního použitého textového dotazu. Nejprve se provede podobnostní vyhledávání (viz Algoritmus 4) za pomoci snímků, od kterých byly vybrány třídy do textového dotazu, a následně se porovná rank hledaného snímku s rankem tohoto snímku ve výsledku textového dotazu, který byl původně použit uživatelem. V případě, že bylo pro podobnostní vyhledávání zaznamenáno více snímků, jsou zpracovány všechny takové snímky a následně je vybrán výsledek, v kterém má hledaný snímek nejnižší rank.

3. Podpůrný software

Podpůrný software byl vytvořen pro sběr dat od uživatelů, evaluaci různých kombinací modelů a zpracování nasbíraných dat v rámci experimentů. Hlavní část softwaru umožňuje vyhledávání v obrázkové databázi pomocí textového dotazu s následným zobrazením výsledků společně s jejich předem definovanými třídami. Tato klasifikace snímků by uživateli podle předpokladu měla pomoci s výběrem vhodných slovních spojení pro formulaci dotazu a následnému pravděpodobnějšímu nalezení hledaného snímku. Kromě textového vyhledávání nástroj umožňuje i podobnostní vyhledávání na základě vybraného obrázku z aktuálně zobrazených, avšak toto vyhledávání je u většiny experimentů skryto, aby neovlivňovalo pozorování testovaných metod. Software umožňuje zaznamenávat průběh vyhledávání jednotlivých uživatelů a měnit nastavení experimentů v rámci změny použitých kolekcí snímků i v rámci změny pozorovaných modelů. Jelikož se jedná pouze o podpůrný software k experimentům bude v této kapitole představen pouze jeho návrh a informace týkající se předzpracování dat prováděné tímto softwarem.

Modely definované v předchozí sekci jsou implementovány přímočaře podle pseudokódů algoritmů zmíněných u jednotlivých modelů. Avšak při vyhledávání uživateli pro sběr dat na experimenty byl použit pouze první model (viz kapitola 2) s fúzí na základě sčítaní a model s inicializací pomocí SOM (viz kapitola 2.3). Ostatní modely jsou v softwaru implementovány, ale jsou použity až pro pozdější evaluaci výsledků experimentů.

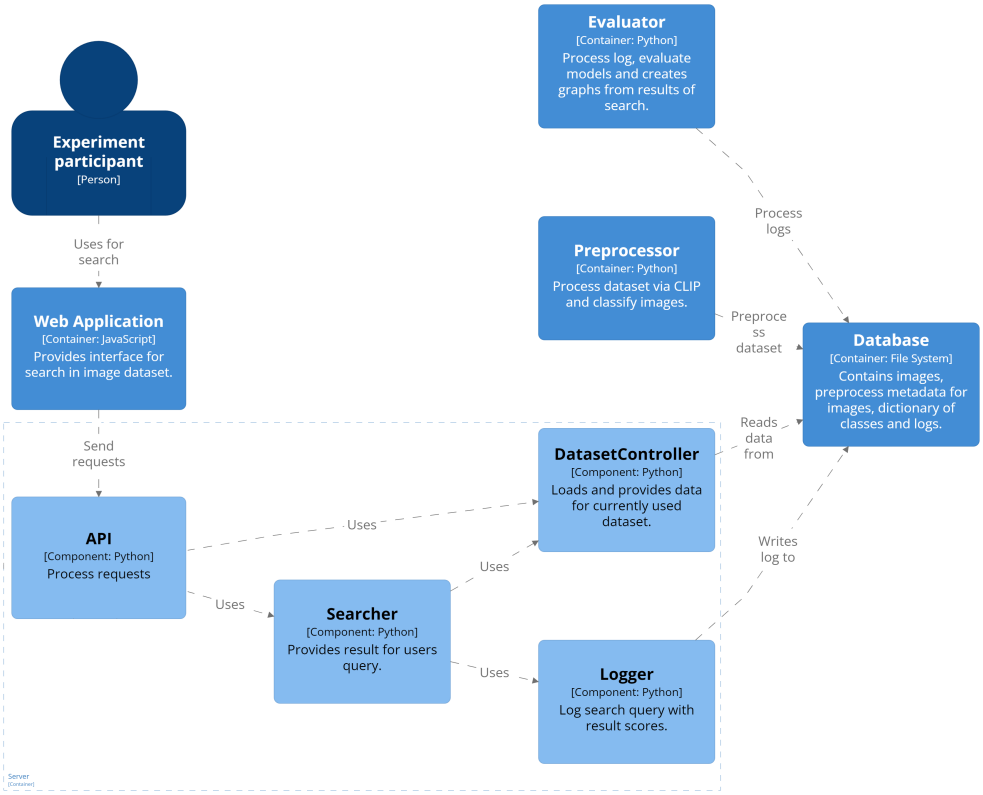
V příloze A je postup na zprovoznění zkušební verze nástroje (na ukázkovém datasetu) společně s odkazy na uživatelskou dokumentaci a podrobnější vývojářskou dokumentaci, která obsahuje i návod na předzpracování vlastního datasetu pro použití softwarem.

3.1 Návrh softwaru

Jak je vidět na Obrázku 3.1, software je složen ze čtyř částí, kterými jsou server a webové rozhraní, preprocesor a evaluátor. Část softwaru, kterou přímo používají uživatelé, je vyvinuta pomocí JavaScriptu jako webové rozhraní, které slouží převážně na zprostředkování komunikace uživatele se serverem.

Serverová část softwaru je vyvinuta pomocí Python knihovny Django (viz kapitola 1.3), která se stará o fungování serveru jako je přijímání dotazů, paralelizace zpracovávání, přesměrovávání a další funkčnosti. Samotná serverová část zpracovává dotazy do obrázkové databáze a odesílá nazpět webovému rozhraní výsledky uživatelských dotazů.

Důležitou součástí softwaru jsou všechna přídavná metadata a samotný dataset, který je složený z předem zpracovaných snímků. Metadata obsahují veškeré předem zpracované snímky v podobě vektorů získaných z neuronové sítě CLIP, seznam tříd klasifikace (i s informacemi o četnostech třídy v používaných kolekcích) a samotné zařazení jednotlivých snímků do tříd.



Obrázek 3.1: Architektura systému.

Preprocesor má na starost zpracování videí na jednotlivé snímky, zpracování snímků pomocí CLIPu a klasifikaci snímků do tříd. Evaluátor se stará o zpracování logu a vyhodnocení výsledků jednotlivých modelů v podobě grafů.

3.1.1 Webové rozhraní

Celé webové rozhraní slouží na interakce s uživatelem, zobrazování výsledků uživatelských dotazů a na zaznamenávání průběhu vyhledávání. Kromě úvodní a koncové obrazovky probíhá celé hledání v jednom okně, které je rozděleno na dvě hlavní části (viz Obrázek 3.2): panel pro vyhledávání v levé části a plochu na zobrazování výsledků.

Panel pro vyhledávání zprostředkovává zadávání textových dotazů uživatelem, zobrazuje nejčastější třídy aktuálně zobrazených snímků a zobrazuje aktuálně vyhledávaný snímek. V ploše pro výsledky je zobrazeno 60 snímků, přičemž u každého snímku je kromě 10 tříd, do kterých je snímek zařazen, možno zobrazit kontext videa (okolní snímky ve videu), z kterého snímek pochází[1]. Každý ze snímků je možné odeslat pro kontrolu shody s hledaným snímkem nebo jako vzor pro podobnostní hledání.

[1]V kontextu je možné se posouvat, ale v rámci experimentu je zakázáno odesílání snímků z kontextu a proto je zamezeno i posouvání.

Obrázek 3.2: Ukázka rozhraní pro vyhledávání nad Marine Video Kit datasetem.

V průběhu vyhledávání je zaznamenáváno přidávání tříd do textového dotazu pomocí tlačítek u jednotlivých snímků. V případě použití třídy v dotazu je snímek, od něhož byla třída vzata, využit na porovnání s podobnostním vyhledáváním (blíže kapitola 2.3).

3.1.2 Server

Při prvotním spuštění serveru jsou do paměti načtena veškerá data aktuálně používané kolekce snímků, aby bylo urychleno uživatelské dotazování. Veškerá data získaná z neuronové sítě CLIP týkající se aktuálně využívané kolekce jsou předzpracována, takže při spouštění serveru se pouze načítají a nejsou na nich prováděné žádné další zpracování.

Díky využití Django je celková struktura kódu serveru určena tímto frameworkem, takže další implementace se stará pouze o zpracování samotného vyhledávání.

Hledání

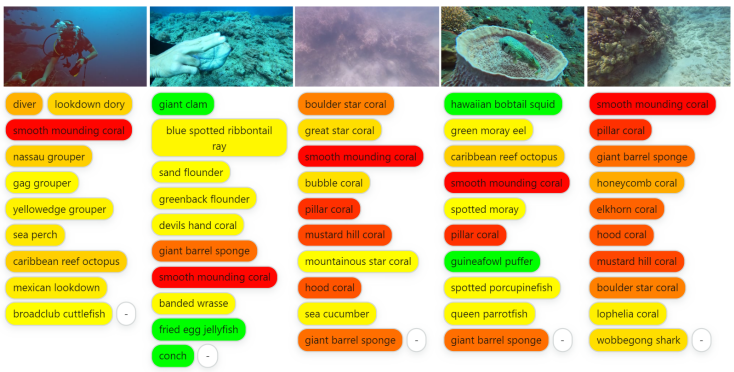
Uživatelské hledání je založeno na textových dotazech. Po odeslání textového dotazu na server je pomocí dat získaných z neuronové sítě CLIP vyhodnocen výsledek hledání v závislosti na používaném modelu (viz algoritmy z kapitoly 2). Po vyhodnocení dotazu jsou data výsledku uložena do logu, jak bude popsáno později. Výsledek je následně zobrazen ve webovém rozhraní jako omezené množství nejrelevantnějších snímků (základní nastavení je 60 snímků), u kterých jsou zobrazeny třídy, do kterých byly snímky zařazeny.

3.2 Zpracování dat

Zpracování obrázkových kolekcí, které jsou používány pro experimenty, je umožněno pomocnými skripty, které souhrnně označíme jako preprocesor. Celkové zpracování datasetu je rozděleno na několik fází, kterými jsou naparsování videí, získání vektorové reprezentace snímků a klasifikace snímků.

3.2.1 Klasifikace snímků

Každý snímek z databáze je zařazen do 10 různých tříd, přičemž slovník tříd je zvolen podle aktuálního datasetu nebo experimentu. Samotné přiřazení tříd[2] je předem vygenerované za pomocí funkce f_{CLIP_W} (viz kapitola 1.2) pro každý snímek. Jak je možné vidět na Obrázku 3.3, třídy jsou barevně odděleny, přičemž jejich barva indikuje frekvenci třídy v aktuální kolekci. Zelenou barvou jsou označeny unikátní třídy, zatímco třídy červené barvy se vyskytují u snímků nejfrekventovaněji.



Obrázek 3.3: Ukázka klasifikace snímků do tříd společně s frekvencí zobrazených tříd v MVK datasetu.

Proces klasifikace snímků se dělí do dvou fází, kde první je získání tříd pro jednotlivé snímky, tedy samotná klasifikace snímků. Druhou fází je získání informací o výskytech jednotlivých tříd v dané kolekci.

Třídy klasifikace

Kategorie, do kterých mohou být jednotlivé snímky zařazeny, jsou rozděleny na dvě skupiny. První skupina obsahuje 6771 nejběžněji užívaných anglických podstatných jmen (získaných z databáze nejčastěji užívaných podstatných jmen[3] a následně protříděných[4]). Tyto kategorie jsou v experimentech použity na kolekcí V3C (blíže zmíněná později), jelikož jsou určeny pro klasifikaci běžných kolekcí snímků. Celkem je ve slovníku obsaženo 6771 různých tříd s frekvencí výskytu do 10,65%. Některé z tříd nejsou pro dataset používaný v experimentu přiřazeny k jakémukoliv snímku, ale ve slovníku jsou zachovány kvůli případnému použití slovníku pro jiný dataset.

Druhá skupina obsahuje názvy podmořských živočichů (získaná z databáze názvů ryb[5], webových stránek MarineBio[6] a Encyklopedie mořského života[7]) doplněné o názvy objektů běžně se vyskytujících pod vodou jako je například vrak

[2]Při zpracování slovníku je použita pro každou třídu předpona "a photo of", která by měla pomoci s lepší klasifikací jednotlivých snímků, jak je navrženo autory CLIPu.[24]

[3]http://www.desiquintans.com/nounlist

[4]Z databáze byla odstraněna některá abstraktní podstatná jména, jako je například uzákonění, nákaza, poradenství nebo uložení.

[5]https://www.mpi.govt.nz/dmsdocument/194-approved-fish-names-list

[6]https://www.marinebio.org/creatures/

[7]https://oceana.org/marine-life/

nebo potápěč. Druhý ze slovníků je použit pro kolekci Marine Video Kit (zmíněnou blíže později), jelikož slouží pouze pro klasifikaci snímků ze specifické kolekce obsahující podmořská videa. V tomto slovníku je celkem 512 tříd, přičemž nejfrekventovanější třída je přiřazena k 70% snímků, což je způsobeno relativně malou velikostí slovníku společně s velkou obecností této třídy. Avšak kromě 12 tříd jsou všechny třídy přiřazeny k méně než 20% snímků.

4. Experimenty

4.1 Testované kolekce dat

Pro experimenty v této práci byly využity dvě kolekce snímků. První množinu tvoří 20000 snímků vybraných z *Vimeo Creative Commons Collection* (neboli zkráceně V3C)[26], která je složena z 28450 videí v celkové délce 3800 hodin. Výběr z kolekce využíval síť TransNet[33], klastrování a následné nasamplování množiny tak jak bylo použito také například v této práci [19].

Druhá množina je složena z podmořských videí z kolekce *Marine Video Kit* (nadále označována zkráceně MVK)[37], z nichž bylo vybráno 22036 snímků (vždy jeden snímek za dvě sekundy). Tato kolekce byla vybrána kvůli obtížnosti formulace dotazů bez znalosti domény podmořských živočichů[37].

4.2 Metodika sběru dat

Pro vyhodnocení efektivity zkoumaných modelů byla nasbírána data od uživatelů, kteří byli seznámeni s funkcionalitou softwaru a průběhem vyhledávání. Všichni uživatelé souhlasili s anonymním sběrem dat během vyhledávání a jejich použitím pro vědecké účely.

Zkoumání vlivu modelu je provedeno pomocí pozorování vývoje ranku hledaného snímku ve výsledku po uživatelských dotazech. Tímto způsobem je sledováno, jestli se rank zlepšuje, a tedy jestli je daná metoda pro uživatele nápomocná. Pro pozorování je nutné zaznamenání textového dotazu uživatele a indexu aktuálně hledaného snímku (viz 4.1). Rozlišení uživatelů je umožněno zaznamenáváním identifikátoru uživatele. Kvůli porovnání modelů s podobnostním vyhledáváním jsou zaznamenávány třídy použité v dotazu (více viz kapitola 2.4). Kromě těchto dat je později ještě vypočítáván nejnižší rank snímků z okolí hledaného snímku ve stejném videu (téměř identické snímky)[1] ve výsledku dotazu.

Tabulka 4.1: Příklady uživatelských dotazů, ve kterých reformulace pomocí tříd výrazně zlepšila pozici hledaného snímku.

Textový dotaz před a po reformulaci	**Pozice**
star	3428
star, great star coral, sea star	18
two rays in the middle, dark sea, sand	4993
two rays in the middle, dark sea, sand, eagle ray, spotted eagle ray	42

Data jsou sbírána pro všechny základní modely najednou, což je možné díky tomu, že uživatel vytváří pouze jednu reformulaci, takže na výsledku ovlivněném druhem modelu již žádný dotaz není vytvářen.

[1] Tato informace je sbírána pouze pro kolekci podmořských videí, jelikož pro snímky z V3C tato informace není potřebná kvůli tomu, že se jedná pouze o podmnožinu kolekce, takže z jednoho videa v kolekci bývá většinově pouze jeden snímek.

4.2.1 Průběh experimentu

Experiment začíná tím, že je uživateli zobrazen náhodný[2] snímek z kolekce, která je aktuálně testovaná, a úkolem uživatele je nalézt zobrazený snímek pouze pomocí textových dotazů. Uživatel k formulaci textových dotazů může[3] využít tříd přiřazených k snímkům zobrazených v aktuálním okně.

Při odesílání textového dotazu jsou zaznamenávány kromě textového dotazu, identifikátoru aktuálně hledaného snímku a identifikátoru uživatele (na začátku náhodně vygenerovaný) i rank aktuálně hledaného snímku ve výsledku hledání a případné třídy použité pro textový dotaz společně s identifikátorem snímku, od něhož byla třída vybrána.

Hledání jednoho ze snímků končí v případě, pokud jej uživatel úspěšně nalezne (i s jeho odesláním na server), pokud uživatel není schopen snímek nalézt ani po dvou textových dotazech (včetně úvodního dotazu) nebo v případě kdy se uživatel rozhodne daný snímek nadále nehledat a zmáčkne tlačítko "Next".

4.3 Rozbor výsledků experimentů

Jelikož je možné většinu modelů kombinovat různými způsoby, nejprve zadefinujeme zkoumané kombinace modelů v následující tabulce. Pro jednoduchost a přehlednost budeme nadále používat zkrácené názvy modelů. U modelů používajících omezení datasetu bude X na konci názvu nahrazeno hodnotou 25, 50 nebo 75 podle používaného procentuálního omezení datasetu. Pro účely vyhodnocení experimentů budeme prvních šest modelů z tabulky 4.2 označovat souhrnně jako *základní modely*.

Pro vyhodnocení efektivity jednotlivých modelů budeme využívat tzv. *houslový graf* (violin plot), který zobrazuje distribuci ranků hledaných snímků ve výsledcích. Na ose x budou zobrazeny jednotlivé názvy aktuálně testovaných modelů a na ose y budou ranky hledaných snímků ve výsledcích vyhodnocených daným modelem. U většiny takovýchto grafů první graf na ose x, obsahující v názvu “1_not_found”, zobrazuje distribuci ranků po prvním uživatelském dotazu. Avšak jak označení napovídá, v těchto datech jsou odfiltrovány výsledky, které byly uživatelem úspěšně nalezeny (zobrazeny do ranku 60)[4].

Pro srovnání některých modelů využijeme ještě graf obsahující kumulativní křivky, které ukazují procento nalezitelných snímků (na vertikální ose) do daného ranku (na horizontální ose).

Celkově se experimentu účastnilo 30 uživatelů, z nichž 15 účastníků vyhledávalo za pomoci klasických modelů a 15 pomocí modelů s inicializací první obrazovky pomocí SOM. Celkově bylo v experimentech hledáno 479 snímků za pomoci klasických modelů a 362 snímků s inicializací pomocí SOM v MVK datasetu. Dále bylo pomocí klasických modelů hledáno 100 snímků nad V3C datasetem.

[2]Pouze část snímků je náhodně vybraná z kolekce, jelikož 10 prvních zobrazených snímků je pro všechny uživatele konstantních kvůli následnému možnému srovnání uživatelů (srovnání kvality textových dotazů i schopnosti nalézt snímek, pokud byl pro uživatele viditelný).

[3]Explicitně bylo uživatelům v úvodu experimentu doporučeno pečlivě se koukat na zobrazené třídy u snímků.

[4]I v případě, že uživatel přímo nenalezl daný snímek, ale jeho rank se nachází v rozmezí do 60, je z dat odfiltrován.

Tabulka 4.2: Přehled zkoumaných modelů

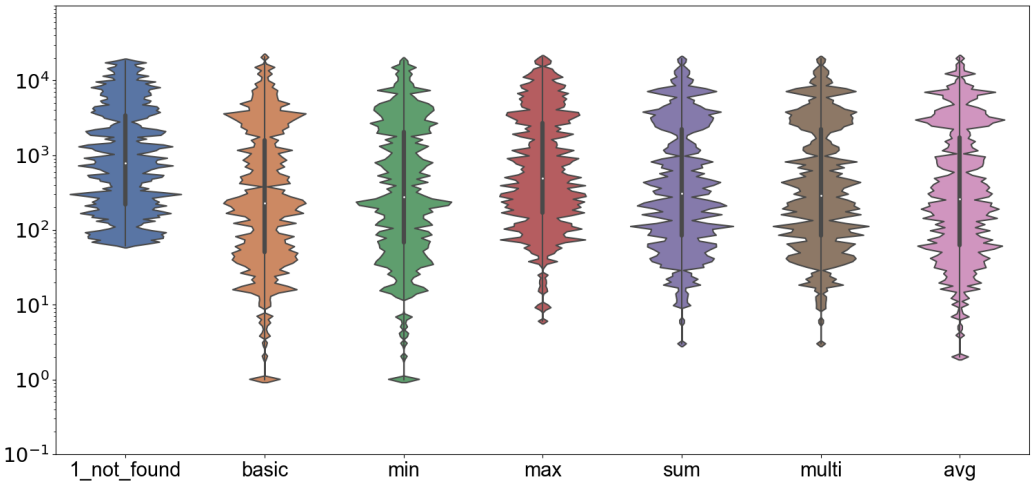
Název modelu	**Použitý model 1. kategorie**	**Omezení datasetu**	**Použití SOM**
basic	základní model bez kombinace	NE	NE
min	fúze za pomoci minima	NE	NE
max	fúze za pomoci maxima	NE	NE
sum	fúze za pomoci sčítaní	NE	NE
multi	fúze za pomoci násobení	NE	NE
avg	fúze za pomoci váženého průměru (druhý dotaz má dvakrát tak větší váhu)	NE	NE
basic_limit_X	základní model bez kombinace	ANO	NE
min_limit_X	fúze za pomoci minima	ANO	NE
max_limit_X	fúze za pomoci maxima	ANO	NE
sum_limit_X	fúze za pomoci sčítaní	ANO	NE
multi_limit_X	fúze za pomoci násobení	ANO	NE
avg_limit_X	fúze za pomoci váženého průměru (druhý dotaz má dvakrát tak větší váhu)	ANO	NE
basic_som	základní model bez kombinace	NE	ANO
min_som	fúze za pomoci minima	NE	ANO
max_som	fúze za pomoci maxima	NE	ANO
sum_som	fúze za pomoci sčítaní	NE	ANO
multi_som	fúze za pomoci násobení	NE	ANO
avg_som	fúze za pomoci váženého průměru (druhý dotaz má dvakrát tak větší váhu)	NE	ANO
basic_som_limit_X	základní model bez kombinace	ANO	ANO
min_som_limit_X	fúze za pomoci minima	ANO	ANO
max_som_limit_X	fúze za pomoci maxima	ANO	ANO
sum_som_limit_X	fúze za pomoci sčítaní	ANO	ANO
multi_som_limit_X	fúze za pomoci násobení	ANO	ANO
avg_som_limit_X	fúze za pomoci váženého průměru (druhý dotaz má dvakrát tak větší váhu)	ANO	ANO

4.3.1 Základní modely

Nejprve začneme vyhodnocením šesti základních modelů na Marine Video Kit a V3C datasetech. Výsledky jednotlivých modelů jsou vidět na Obrazcích 4.1 a 4.2.

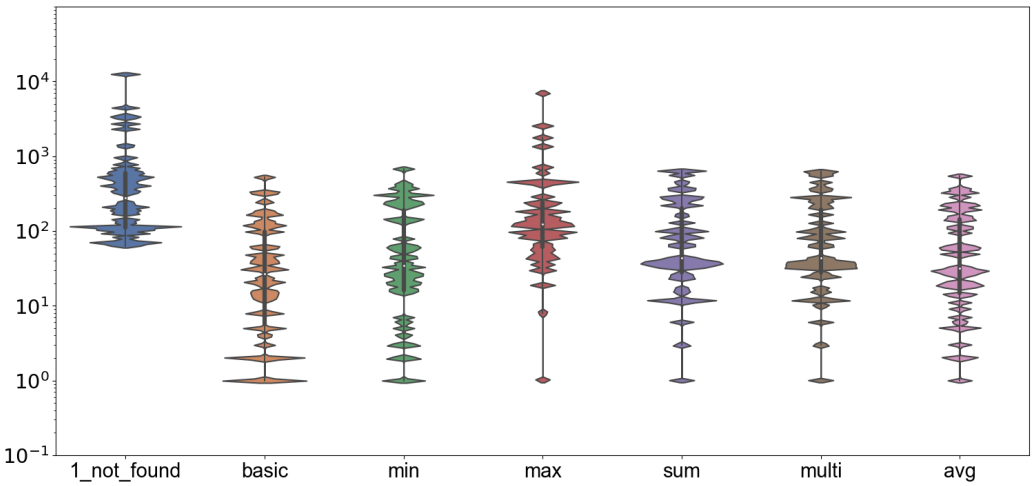
Pro tyto modely bylo prvním dotazem nalezeno 146 snímků (skoro 30.5% z hledaných) v MVK datasetu a 54 snímků (54% z hledaných) v případě V3C. Velké procento nalezených snímků již prvním dotazem ve V3C datasetu je nejspíše způsobeno jednodušší formulovatelností dotazů uživatelem, jelikož se jedná o velmi

obecný dataset. Ale také tím, že používaný dataset je obsahově velmi různorodý, tudíž již při relativně nepřesné popisu je velká část snímků dohledatelná.



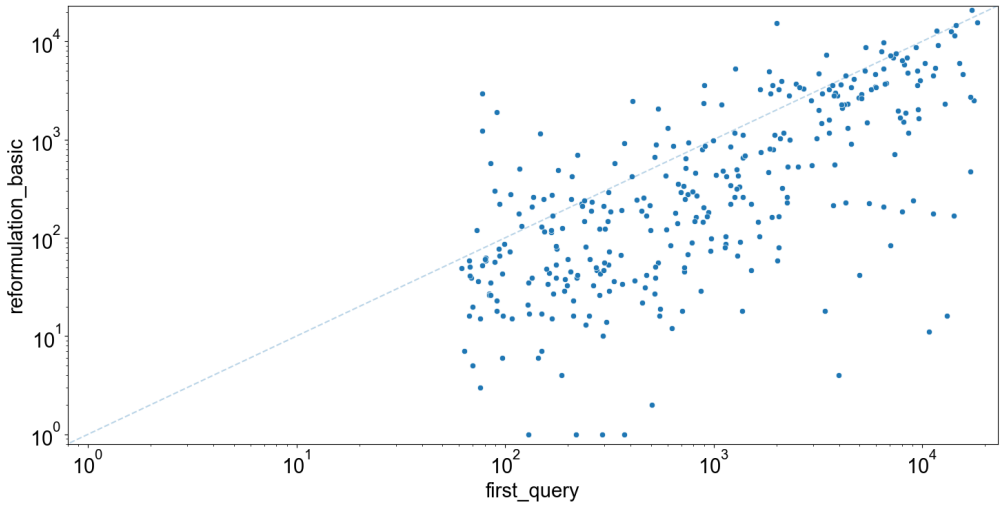
Obrázek 4.1: Výsledky základních modelů na Marine Video Kit datasetu.

V obou datasetech nejlépe vychází basic model, který nepoužívá při reformulaci přímo výsledky prvního dotazu. Tento překvapivý výsledek může být způsoben i převládajícími výrazně lepšími reformulovanými dotazy, které již používají jednotlivé třídy. Toto pozorování je podpořeno i faktem, že také modely sum a multi, které využívají oba dotazy se stejnou váhou, dopadají na obou datasetech výrazně hůře. Pro basic model je možné toto zlepšení reformulovaných dotazů oproti prvotnímu dotazu pozorovat na grafu 4.3. Reformulace textového dotazu pomocí tříd snímků v tomto experimentu dokonce snížila rank hledaného snímku v 277 případech (83% ze všech hledaných).



Obrázek 4.2: Výsledky základních modelů na V3C datasetu.

Jak je vidět z grafů 4.1 a 4.2, obecně všechny modely snižují medián ranků hledaných snímků ve výsledcích po reformulaci.

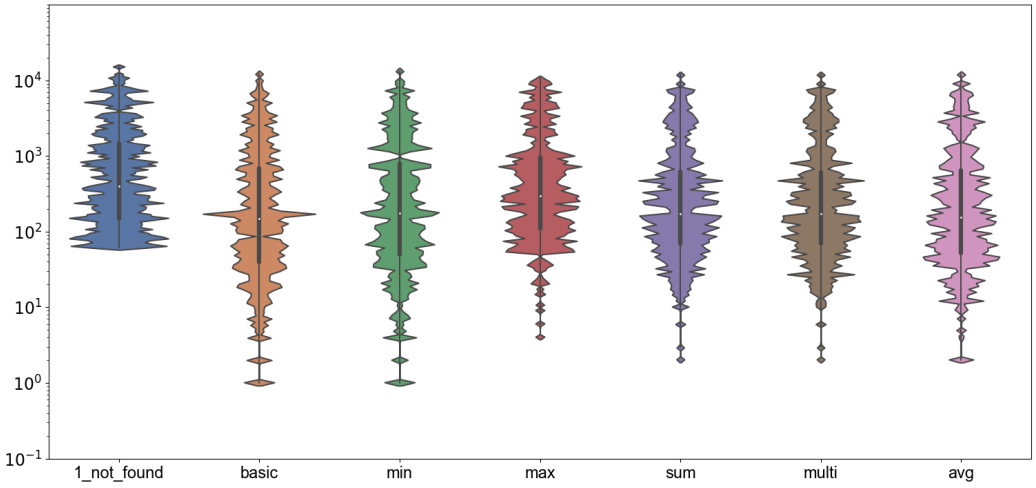


Obrázek 4.3: Srovnání ranku hledaných snímků po prvním a druhém reformulovaném textovém dotazu s použitím basic modelu v MVK datasetu.

Nalezitelnost snímků

Kvůli evaluaci jednotlivých modelů je vyžadováno po uživatelích nalezení konkrétního snímku, avšak v MVK datasetu, stejně jako v mnoha dalších datasetech složených z videí, jsou okolní snímky téměř totožné a díky tomu je jednoduché některé snímky dohledat i pomocí jejich okolí ve videu. Pro problém samotného vyhledávání ve videu je tedy důležité se podívat i na možnou dohledatelnost snímku právě pomocí zobrazení okolí snímku. Na grafu 4.4 je vidět distribuce nejnižšího ranku hledaného snímku a jeho těsného okolí ve videu (dva vedlejší snímky z obou stran pro každý snímek).

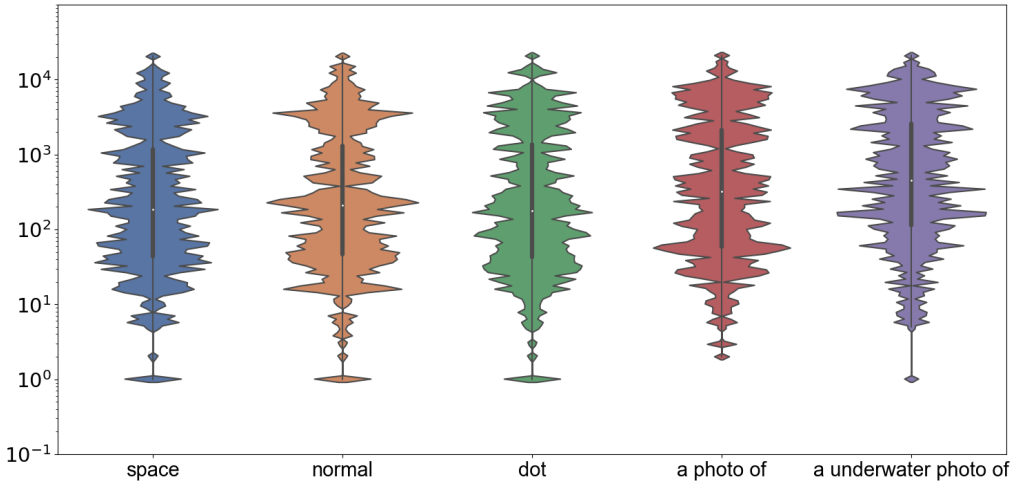
V tomto případě je vidět, že výsledky jsou opět lepší než po prvním dotazu, z něhož se též odebraly snímky, které bylo možné dohledat pomocí jejich okolí. Celkově těmito modely bylo po prvním dotazu nalezitelných (daný snímek nebo jeho těsné okolí má rank do 60) dokonce 200 snímků (skoro 42% z hledaných). Při použití basic modelu bylo nalezitelných 93 snímků, což je 33% ze všech hledaných snímků při reformulaci. Pro srovnání, v grafu 4.1 bylo přímo po reformulaci basic modelu nalezeno do ranku 60 také 93 snímků, ale celkový počet dohledatelných snímků (po prvním dotazu i po reformulaci) pomocí basic modelu byl 239 (skoro 50%), zatímco při započtení okolí snímku bylo dohledatelných 293 snímků (61% z hledaných). Tudíž započtení okolí snímku nijak nepřekvapivě pomáhá s lepší dohledatelností jednotlivých snímků.



Obrázek 4.4: Výsledky ranků okolních snímků základních modelů na Marine Video Kit datasetu.

Vliv způsobu formulace

V práci jsou používány pro připojení nových tříd do dotazu čárky, avšak jak zmiňují tvůrci neuronové sítě CLIP v jejich článku [24], formulace textových dotazů má relativně velký vliv na výsledný vektor. Proto jsme zkusili otestovat i jiné způsoby formulace dotazů, zahrnující změnu všech spojení (nahrazeny jsou všechny čárky ne pouze ty připojující třídy) a také přidání "a photo of" na začátek dotazu (inspirováno zmíněným článkem).



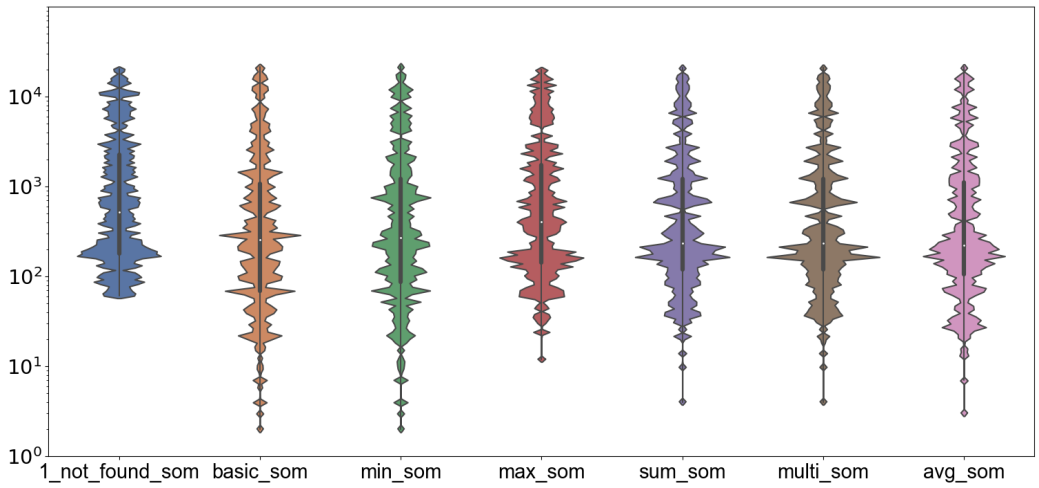
Obrázek 4.5: Výsledky basic modelu pro různé formulace na Marine Video Kit datasetu.

V grafu 4.5 jsou vidět výsledky jednotlivých změn formulací pro basic model v Marine Video Kit datasetu, přičemž původní dotaz je označen jako normal. Nejlepší výsledky s použitím basic modelu překvapivě dosáhla formulace pouze pomocí mezer. Použití teček přineslo o trochu horší výsledek a nejhůře z těchto tří

modelů dopadla původní formulace. Co se týče přidání pojmenování fotky na začátek dotazu, tak jak obecnější "a photo of", tak přesnější "a underwater photo of" dopadlo hůře než model, který tyto označení nepoužívá. Výrazné zhoršení modelu při použití "a underwater photo of" oproti původnímu dotazu je nejspíše způsobeno tím, že modely jsou vyhodnocovány nad MVK datasetem, v kterém všechny snímky jsou pořizovány pod vodou.

4.3.2 Vliv použití SOM

Pro tyto modely bylo prvním dotazem nalezeno celkem 123 snímků, což je procentuálně mírně větší množství[5] než bez použití inicializované první obrazovky. Navíc medián ranků hledaných snímků po prvním dotazu (včetně snímků s rankem pod 60) s použitím SOM vyšel roven 203.5, zatímco bez použití SOM (základní modely) byl roven 274. Toto snížení podporuje hypotézu, že zobrazované třídy snímků pomáhají s lepší formulací textových dotazů. Výsledky jednotlivých modelů po reformulaci používajících SOM pro inicializaci úvodní obrazovky jsou vidět na obrázku 4.6.



Obrázek 4.6: Výsledky modelů používajících SOM na Marine Video Kit datasetu.

V tomto případě je těžší jednoznačně označit, který z modelů dopadl obecně nejlépe, protože i když basic_som model má nejnižší dolní kvartil a tedy nejvíce přímo dohledatelných snímků, tak nemá nejnižší medián (možno pozorovat jak v tabulce 4.4, tak na grafu 4.6). Z pohledu nejnižšího mediánu v tomto případě dopadl nejlépe avg_som model. Na tom, že i modely multi_som a sum_som z hlediska mediánu dopadly lépe než basic_som model, je možné pozorovat, že je při použití SOM pravděpodobně o dost důležitější i první dotaz, který právě basic_som zanedbává.

[5]Bez použití SOM bylo nalezeno 30,5%, zatímco s použitím SOM bylo nalezeno po prvním dotazu 34% hledaných snímků.

4.3.3 Omezování datasetu

V případě omezování datasetu jsou výsledky kvůli přehlednosti zobrazeny pomocí tabulky, jelikož při omezení se může hledaný snímek dostat pryč z datasetu, ve kterém se aktuálně vyhledává. V tomto případě není možné zjistit reálnou pozici hledaného snímku v datasetu, jelikož jeho pozice již není dostupná a tak se nachází v množině všech snímků, které byly zanedbány.

V tabulkách 4.3 a 4.4 jsou znázorněny mediány a dolní kvartily jednotlivých modelů (společně se základními modely kvůli porovnání vztahů jednotlivých modelů ze skupiny) a také počet dotazů, které nebylo možné dohledat, jelikož byly při reformulaci v části datasetu, která byla zanedbána. Medián i dolní kvartil je vypočítán pouze z ranků snímků, které byly dohledatelné a proto není možné srovnání napříč skupinami modelů.

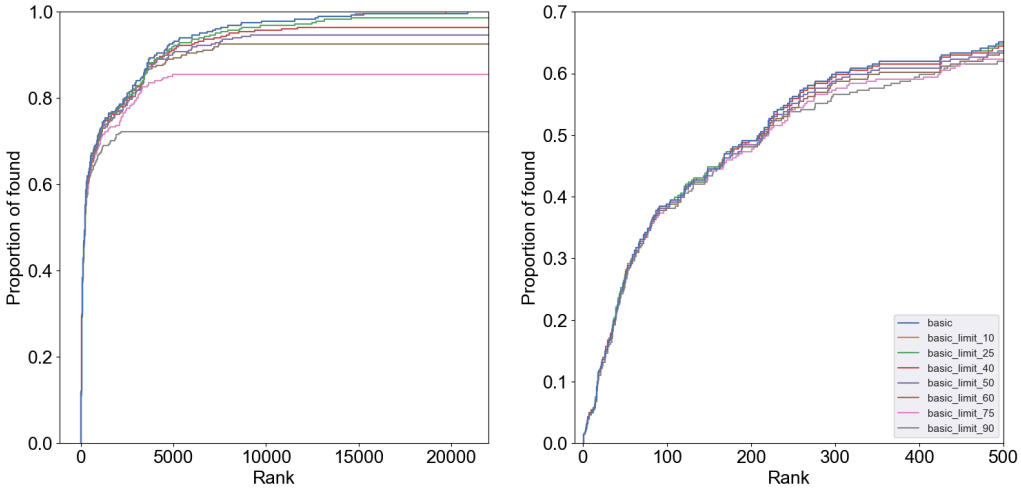
Tabulka 4.3: Výsledky omezování datasetu pro klasické modely na MVK datasetu

Název modelu	**Medián**	**Dolní kvartil**	**Počet mimo dataset**
basic	227.0	51.0	0
min	275.0	68.0	0
max	491.0	170.0	0
sum	308.0	85.0	0
multi	292.0	85.0	0
avg	263.0	63.0	0
basic_limit_25	222.5	50.0	5
min_limit_25	260.0	67.0	5
max_limit_25	481.0	168.0	5
sum_limit_25	278.0	84.75	5
multi_limit_25	272.0	84.75	5
avg_limit_25	241.0	62.75	5
basic_limit_50	220.0	48.5	18
min_limit_50	243.0	65.0	18
max_limit_50	418.0	162.0	18
sum_limit_50	243.0	80.0	18
multi_limit_50	243.0	80.0	18
avg_limit_50	219.0	60.5	18
basic_limit_75	167.5	43.0	53
min_limit_75	209.5	56.0	53
max_limit_75	334.0	128.5	53
sum_limit_75	200.5	65.0	53
multi_limit_75	198.5	64.75	53
avg_limit_75	176.5	55.75	53

Jak je vidět v obou tabulkách, obecně i po omezení datasetu zůstávají převážně nejlepšími model basic mezi základními modely a model avg_som v případě

použití SOM.

Graf 4.7 ukazuje kumulativní počet hledání snímků, kdy byl snímek nalezen po daný rank na ose x. Je zde patrné, že modely omezující dataset mají své limity v počtu dohledatelných snímků, obzvláště pro vyšší rank těchto snímků ve výsledku.



Obrázek 4.7: Srovnání basic modelů s různými omezeními Marine Video Kit datasetu. V grafu jsou zahrnuty i nenalezitelné snímky.

Rizikem omezování datasetu je hlavně případ, kdy snímky nemusí být již dohledatelné vůbec. Toto riziko je samozřejmě větší s rostoucím procentuálním množstvím, kterým se omezuje dataset. Jak je vidět z obou tabulek, tak pro modely, ve kterých se zanedbá 75% datasetu, není možné nalézt dokonce 16% snímků. Avšak při zanedbání 25% není nalezitelných pouze kolem 1% snímků, což už je relativně zanedbatelné množství a proto takové omezení je možné využívat například pro velké datasety, ve kterých takové omezení může pomoci s výkonností systému.

Kromě snížení výpočetní náročnosti je výhodou těchto modelů také skutečnost, že při dobrém prvním dotazu[6] je odebráno velké množství irelevantních snímků. Díky tomu se při druhém dotazu hledaný snímek již nemůže tak výrazně vzdálit, jak by se to mohlo stát bez použití omezení datasetu, i v případě horšího dotazu. To je možné pozorovat i na pravém podgrafu 4.7, kde modely mírně omezující dataset dokonce v okolí ranku 150 dopadají malinko lépe. Nejedná se však celkově o výrazné zlepšení.

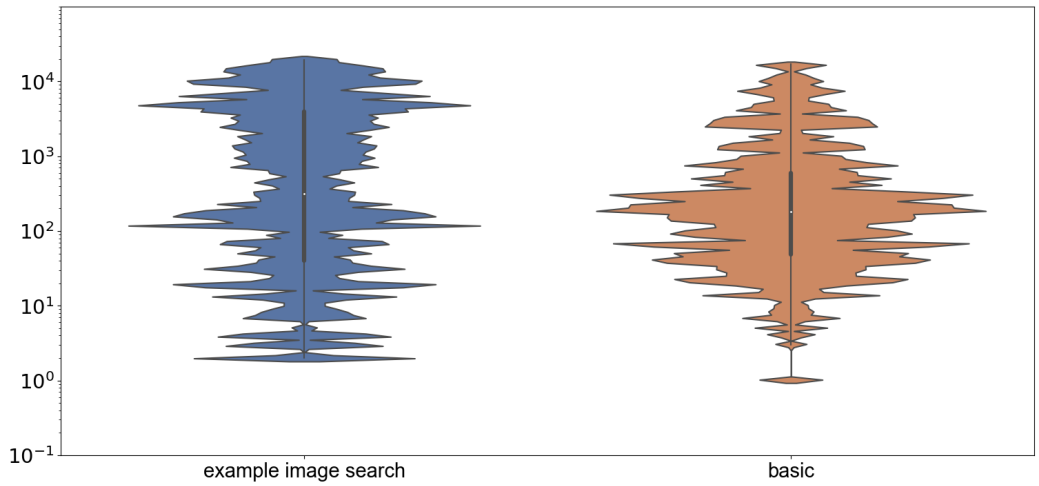
[6]Jedná se o dotaz, po kterém omezením datasetu není hledaný snímek odfiltrován.

Tabulka 4.4: Výsledky omezování datasetu pro modely používající SOM na MVK datasetu

Název modelu	**Medián**	**Dolní kvartil**	**Počet mimo dataset**
basic_som	255.0	69.5	0
min_som	271.0	87.5	0
max_som	409.0	142.5	0
sum_som	233.0	120.0	0
multi_som	236.0	119.5	0
avg_som	221.0	105.5	0
basic_som_limit_25	244.5	69.0	3
min_som_limit_25	258.0	85.25	3
max_som_limit_25	396.0	137.5	3
sum_som_limit_25	228.0	116.75	3
multi_som_limit_25	226.0	115.75	3
avg_som_limit_25	217.0	104.5	3
basic_som_limit_50	223.5	68.0	13
min_som_limit_50	244.5	80.0	13
max_som_limit_50	365.0	134.25	13
sum_som_limit_50	220.0	111.0	13
multi_som_limit_50	219.5	110.5	13
avg_som_limit_50	209.0	93.5	13
basic_som_limit_75	195.0	64.5	37
min_som_limit_75	217.5	71.0	37
max_som_limit_75	292.0	126.5	37
sum_som_limit_75	200.0	98.5	37
multi_som_limit_75	200.0	98.75	37
avg_som_limit_75	181.0	77.75	37

4.3.4 Srovnání s hledáním pomocí snímku

Vzhledem k tomu, že v předchozích experimentech obecně nejlépe dopadl basic model, tak budeme srovnávat pouze tento model s podobnostním vyhledáváním podrobněji popsaném v kapitole 2.4. Je důležité zmínit, že se jedná pouze o porovnání s jedním konkrétním způsob podobnostního vyhledávání, navíc s využitím jednoho specifického modelu neuronové sítě CLIP. V grafu na obrázku 4.8 jsou zobrazeny výsledky dotazů, které byly posbírány způsobem jak je popsáno v kapitole 2.4 při hledání nad Marine Video Kit datasetem. Jelikož toto podobnostní vyhledávání ani basic model nevyužívají přímo data z předchozího vyhledávání, využili jsme této nezávislosti a zkombinovali jsme dohromady data získaná z vyhledávání pomocí modelů basic a basic_som. Tedy v následovném grafu basic označuje výsledky hledání obou zmíněných modelů.



Obrázek 4.8: Srovnání basic modelu s podobnostním vyhledáváním pomocí obrázku jako příkladu na Marine Video Kit datasetu.

I přesto, že na první pohled je vidět, že pomocí tohoto podobnostního vyhledávání je možné nalézt více snímků v zobrazeném výsledku (snímky do ranku 60), basic model má nižší medián a tedy průměrně hledaný snímek skončí na nižší pozici (183.0 pro basic model a 312.5 v případě tohoto podobnostního vyhledávání). Naše hypotéza je, že toto podobnostní vyhledávání pomocí CLIPu funguje lépe v případě, kdy máme k dispozici velmi podobné snímky. Avšak pro případy, kdy snímek obsahuje pouze podobný objekt, metoda použití tříd může vycházet lépe. Na grafy má také vliv aproximace výběru obrázků pro podobnostní hledání.

Závěr

V této práci jsme zkoumali účinnost interaktivního vyhledávacího systému v kombinaci s klasifikací snímků generovanou pomocí neuronové sítě CLIP. Na základě našich experimentů s datasety V3C a Marine Video Kit jsme dospěli k závěru, že tato metodika je užitečná především pro uživatele s omezenou slovní zásobou v dané doméně nebo obecněji pro vyhledávání ve velmi specifických datasetech, jakým je třeba Marine Video Kit dataset. Tato strategie také pomáhá zmenšovat sémantickou mezeru mezi vnímáním uživatele a neuronové sítě CLIP.

Naše výsledky ukázaly, že bez inicializace pomocí SOM je z testovaných modelů nejúčinnější model, který využívá pouze výsledky reformulace, což je možné vysvětlit častou lepší reformulací dotazu díky třídám snímků. Zatímco při použití SOM jsou obecně lepší modely, které berou v potaz i výsledky prvního dotazu, na čemž je vidět užitečnost tříd pro formulaci dotazu.

Také jsme předběžně porovnávali vybranou metodu s podobnostním vyhledáváním pomocí snímku jako příkladu. Podobnostní hledání pomocí obrázku fungovalo lépe v případech, kdy uživatel hledá snímky do ranku 60, zatímco metoda použití tříd snímků vedla ke zlepšení mediánu. Navíc jsme představili pomocný software, který umožňuje sběr dat pro experimenty a jejich následné vyhodnocení.

Na základě prvních výsledků této práce, které jsou podrobněji popsány v tomto článku [15], byl přidán basic model do nástroje CVHunter na již uskutečněné soutěži VBS2023.

Do budoucna bychom rádi zdokonalili kombinaci automatické klasifikace s dalšími klasickými metodami vyhledávání ve videu, jako jsou temporální dotazy nebo zpětnovazební učení. Také bychom chtěli provést lepší prozkoumání vlivu vícenásobné reformulace na využití tříd snímků. A v neposlední řadě porovnat modely s jinými druhy podobnostního vyhledávání než je podobnostní vyhledávání s využitím CLIPu zmíněné v této práci.

Seznam použité literatury

[1] Django home page. URL https://www.djangoproject.com/.

[2] (2022). *LSC '22: Proceedings of the 5th Annual on Lifelog Search Challenge*, New York, NY, USA. Association for Computing Machinery. ISBN 9781450392396.

[3] Amato, G., Bolettieri, P., Carrara, F., Falchi, F., Gennaro, C., Messina, N., Vadicamo, L. a Vairo, C. (2023). Visione at video browser showdown 2023. In Dang-Nguyen, D.-T., Gurrin, C., Larson, M., Smeaton, A. F., Rudinac, S., Dao, M.-S., Trattner, C. a Chen, P., editors, *MultiMedia Modeling*, pages 615–621, Cham, 2023. Springer International Publishing. ISBN 978-3-031-27077-2.

[4] Barthel, K. U., Hezel, N., Jung, K. a Schall, K. (2023). Improved evaluation and generation of grid layouts using distance preservation quality and linear assignment sorting. *Computer Graphics Forum*, **42**(1), 261–276. doi: https://doi.org/10.1111/cgf.14718. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14718.

[5] Cox, I., Miller, M., Minka, T., Papathomas, T. a Yianilos, P. (2000). The bayesian image retrieval system, pichunter: theory, implementation, and psychophysical experiments. *IEEE Transactions on Image Processing*, **9**(1), 20–37. doi: 10.1109/83.817596.

[6] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. a Fei-Fei, L. (2009). ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[7] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. a Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale.

[8] Fang, H., Xiong, P., Xu, L. a Chen, Y. (2021). Clip2video: Mastering video-text retrieval via image clip.

[9] Forcier, J., Bissex, P. a Chun, W. J. (2008). *Python web development with Django*. Addison-Wesley Professional.

[10] Gurrin, C., Þór Jónsson, B., Schöffmann, K., Dang-Nguyen, D.-T., Lokoč, J., Tran, M.-T., Hürst, W., Rossetto, L. a Healy, G. (2023). Introduction to the fifth annual lifelog search challenge, lsc'23. In *Proc. International Conference on Multimedia Retrieval (ICMR'23)*, Thessaloniki, Greece, 2023. ACM.

[11] Hare, J. S., Lewis, P. H., Enser, P. G. B. a Sandom, C. J. (2006). Mind the gap: another look at the problem of the semantic gap in image retrieval. In Chang, E. Y., Hanjalic, A. a Sebe, N., editors, *Multimedia Content Analysis, Management, and Retrieval 2006*, volume 6073, page 607309. International Society for Optics and Photonics, SPIE. doi: 10.1117/12.647755. URL https://doi.org/10.1117/12.647755.

[12] Hezel, N. a Barthel, K. U. (2018). Dynamic construction and manipulation of hierarchical quartic image graphs. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, ICMR '18, page 513–516, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450350464. doi: 10.1145/3206025.3206093. URL https://doi.org/10.1145/3206025.3206093.

[13] Kohonen, T. (1990). The self-organizing map. *Proceedings of the IEEE*, **78**(9), 1464–1480. doi: 10.1109/5.58325.

[14] Lokoč, J., Mejzlík, F., Souček, T., Dokoupil, P. a Peška, L. (2022). Video search with context-aware ranker and relevance feedback. In Þór Jónsson, B., Gurrin, C., Tran, M.-T., Dang-Nguyen, D.-T., Hu, A. M.-C., Huynh Thi Thanh, B. a Huet, B., editors, *MultiMedia Modeling*, pages 505–510, Cham, 2022. Springer International Publishing. ISBN 978-3-030-98355-0.

[15] Lokoč, J., Vopálková, Z., Dokoupil, P. a Peška, L. (2023). Video search with clip and interactive text query reformulation. In Dang-Nguyen, D.-T., Gurrin, C., Larson, M., Smeaton, A. F., Rudinac, S., Dao, M.-S., Trattner, C. a Chen, P., editors, *MultiMedia Modeling*, pages 628–633, Cham, 2023. Springer International Publishing. ISBN 978-3-031-27077-2.

[16] Lokoč, J., Veselý, P., Mejzlík, F., Kovalčík, G., Souček, T., Rossetto, L., Schoeffmann, K., Bailer, W., Gurrin, C., Sauter, L., Song, J., Vrochidis, S., Wu, J. a Jónsson, B. t. (2021). Is the reign of interactive search eternal? findings from the video browser showdown 2020. *ACM Trans. Multimedia Comput. Commun. Appl.*, **17**(3). ISSN 1551-6857. doi: 10.1145/3445031. URL https://doi.org/10.1145/3445031.

[17] Lokoč, J. a Peška, L. (2023). A study of a cross-modal interactive search tool using clip and temporal fusion. In *MultiMedia Modeling - 29th International Conference, MMM 2023, Bergen, Norway, January 9-12, 2023*, Lecture Notes in Computer Science. Springer.

[18] Ma, Z., Wu, J., Loo, W. a Ngo, C.-W. (2023). Reinforcement learning enhanced pichunter for interactive search. In Dang-Nguyen, D.-T., Gurrin, C., Larson, M., Smeaton, A. F., Rudinac, S., Dao, M.-S., Trattner, C. a Chen, P., editors, *MultiMedia Modeling*, pages 690–696, Cham, 2023. Springer International Publishing. ISBN 978-3-031-27077-2.

[19] Mejzlík, F. (2020). Evaluace vyhledávacích modelů založených na klíčových slovech pro hledání známých scén. URL https://dspace.cuni.cz/handle/20.500.11956/119427.

[20] Messina, N., Stefanini, M., Cornia, M., Baraldi, L., Falchi, F., Amato, G. a Cucchiara, R. (2022). Aladin: Distilling fine-grained alignment scores for efficient image-text matching and retrieval.

[21] O'Neil, E. J. (2008). Object/relational mapping 2008: Hibernate and the entity data model (edm). In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1351–1356, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605581026. doi: 10.1145/1376616.1376773. URL https://doi.org/10.1145/1376616.1376773.

[22] Peška, L., Kovalčík, G., Souček, T., Škrhák, V. a Lokoč, J. (2021). W2vv++ bert model at vbs 2021. In Lokoč, J., Skopal, T., Schoeffmann, K., Mezaris, V., Li, X., Vrochidis, S. a Patras, I., editors, *MultiMedia Modeling*, pages 467–472, Cham, 2021. Springer International Publishing. ISBN 978-3-030-67835-7.

[23] Pouyanfar, S., Yang, Y., Chen, S.-C., Shyu, M.-L. a Iyengar, S. S. (2018). Multimedia big data analytics: A survey. *ACM Comput. Surv.*, **51** (1). ISSN 0360-0300. doi: 10.1145/3150226. URL https://doi.org/10.1145/3150226.

[24] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G. a Sutskever, I. (2021). Learning transferable visual models from natural language supervision. URL https://arxiv.org/abs/2103.00020.

[25] Rossetto, L. (2018). *Multi-modal video retrieval*. PhD thesis, University of Basel, Faculty of Science.

[26] Rossetto, L., Schuldt, H., Awad, G. a Butt, A. A. (2018). V3c - a research video collection. URL https://arxiv.org/abs/1810.04401.

[27] Rossetto, L., Gasser, R., Heller, S., Parian-Scherb, M., Sauter, L., Spiess, F., Schuldt, H., Peška, L., Souček, T., Kratochvíl, M., Mejzlík, F., Veselý, P. a Lokoč, J. (2021). On the user-centric comparative remote evaluation of interactive video search systems. *IEEE MultiMedia*, **28**(4), 18–28. doi: 10.1109/MMUL.2021.3066779.

[28] Rossetto, L., Gasser, R., Lokoč, J., Bailer, W., Schoeffmann, K., Muenzer, B., Souček, T., Nguyen, P. A., Bolettieri, P., Leibetseder, A. a Vrochidis, S. (2021). Interactive video retrieval in the age of deep learning – detailed evaluation of vbs 2019. *IEEE Transactions on Multimedia*, **23**, 243–256. doi: 10.1109/TMM.2020.2980944.

[29] Schall, K., Hezel, N., Jung, K. a Barthel, K. U. (2023). Vibro: Video browsing with semantic and visual image embeddings. In Dang-Nguyen, D.-T., Gurrin, C., Larson, M., Smeaton, A. F., Rudinac, S., Dao, M.-S., Trattner, C. a Chen, P., editors, *MultiMedia Modeling*, pages 665–670, Cham, 2023. Springer International Publishing. ISBN 978-3-031-27077-2.

[30] Schoeffmann, K., Lokoč, J. a Bailer, W. (2021). 10 years of video browser showdown. In *Proceedings of the 2nd ACM International Conference on Multimedia in Asia*, MMAsia '20, New York, NY, USA, 2021. Association

for Computing Machinery. ISBN 9781450383080. doi: 10.1145/3444685.3450215. URL https://doi.org/10.1145/3444685.3450215.

[31] Singhal, A. a kol. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull.*, **24**(4), 35–43.

[32] Smeulders, A., Worring, M., Santini, S., Gupta, A. a Jain, R. (2000). Content-based image retrieval at the end of the early years. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **22**(12), 1349–1380. doi: 10.1109/34.895972.

[33] Souček, T., Moravec, J. a Lokoč, J. (2019). Transnet: A deep network for fast detection of common shot transitions. URL https://arxiv.org/abs/1906.03363.

[34] Spiess, F., Heller, S., Rossetto, L., Sauter, L., Weber, P. a Schuldt, H. (2023). Traceable asynchronous workflows in video retrieval with vitrivr-vr. In Dang-Nguyen, D.-T., Gurrin, C., Larson, M., Smeaton, A. F., Rudinac, S., Dao, M.-S., Trattner, C. a Chen, P., editors, *MultiMedia Modeling*, pages 622–627, Cham, 2023. Springer International Publishing. ISBN 978-3-031-27077-2.

[35] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. a Rabinovich, A. (2014). Going deeper with convolutions.

[36] Tan, M. a Le, Q. V. (2020). Efficientnet: Rethinking model scaling for convolutional neural networks.

[37] Truong, Q.-T., Vu, T.-A., Ha, T.-S., Lokoc, J., Tim, Y. H. W., Joneja, A. a Yeung, S.-K. (2022). Marine video kit: A new marine video dataset for content-based analysis and retrieval.

[38] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u. a Polosukhin, I. (2017). Attention is all you need. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S. a Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[39] Veselý, P., Mejzlík, F. a Lokoč, J. (2021). Somhunter v2 at video browser showdown 2021. In Lokoč, J., Skopal, T., Schoeffmann, K., Mezaris, V., Li, X., Vrochidis, S. a Patras, I., editors, *MultiMedia Modeling*, pages 461–466, Cham, 2021. Springer International Publishing. ISBN 978-3-030-67835-7.

[40] Ying-jie, Z. a Xue-feng, Z. (2018). Review of pattern driven software architecture design [j]. *Computer Science*, **45**(S2), 48–52.

Seznam obrázků

A. Přílohy

A.1 Uživatelská dokumentace

Pro zprovoznění podpůrného softwaru je nutné mít nainstalovaný Python 3. Dataset připojený přímo k této práci slouží pouze pro demonstraci funkčnosti systému. Také je dočasně dostupná online verze webového rozhraní, ve které je možné si vyzkoušet vyhledávání nad Marine Video Kit datasetem s inicializací úvodní obrazovky pomocí SOM.

Veškerá uživatelská dokumentace je dostupná u samotného projektu. Rozšíření nástroje a připojení nového datasetu je popsáno v programátorské dokumentaci, která je též připojena k samotnému projektu. Návod na spuštění nástroje je uveden níže, stejně tak v samotné dokumentaci.

A.1.1 Požadavky na spuštění

- Python 3[1]
- volný port 8000

A.1.2 Sestavení a spuštění

```
# naklonování repozitáře
git clone https://gitlab.mff.cuni.cz/vopalkoz/term-project.git

cd term-project\gasearcher

# spuštění ukázkového modulu
# na windows
start_server.bat

# na linuxu
chmod +x start_server.sh
./start_server.sh
```

V případě, že v průběhu nevznikla nějaká chyba, tak je možné aplikaci nalézt na http:\\localhost:8000.

A.2 Vývojářská dokumentace

Vývojářskou dokumentaci k celému podpůrnému softwaru[2], který je pracovně nazván GASearcher, je možné nalézt ve složce docs. Tato dokumentace obsahuje také návod pro zpracování nových dat. Jelikož většina složitější logiky je schována v samotných modelech, které jsou popsány v této práci, samotná dokumentace je celkem stručná.

[1] https://www.python.org/downloads/

[2] https://gitlab.mff.cuni.cz/vopalkoz/term-project